BASE DE DONNEES

1. Modélisation BDD

I. Introduction

Les sociétés se développent = volume de données de plus en plus important

Sources de données :

-commerce et industrie

-sciences

-société

Pourquoi stocker des données ?

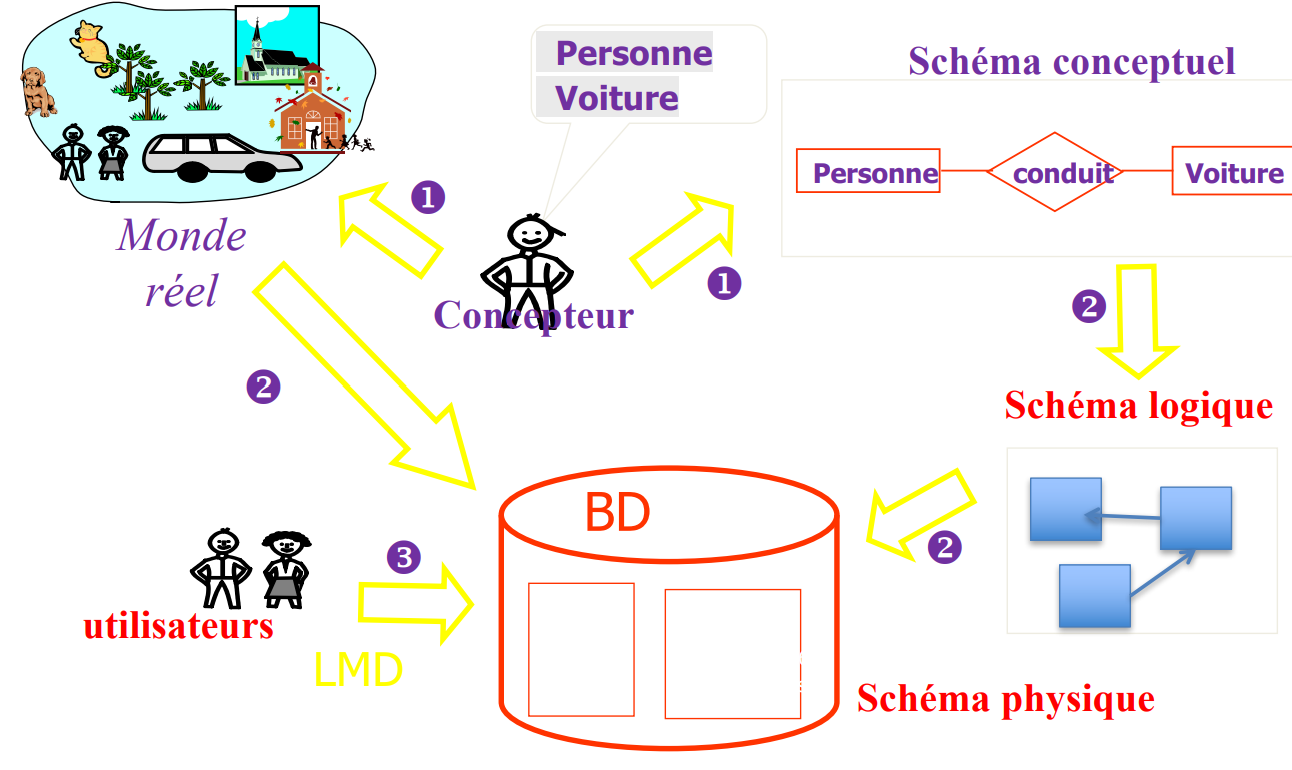
-analyse de contenu

-analyse décisionnelle

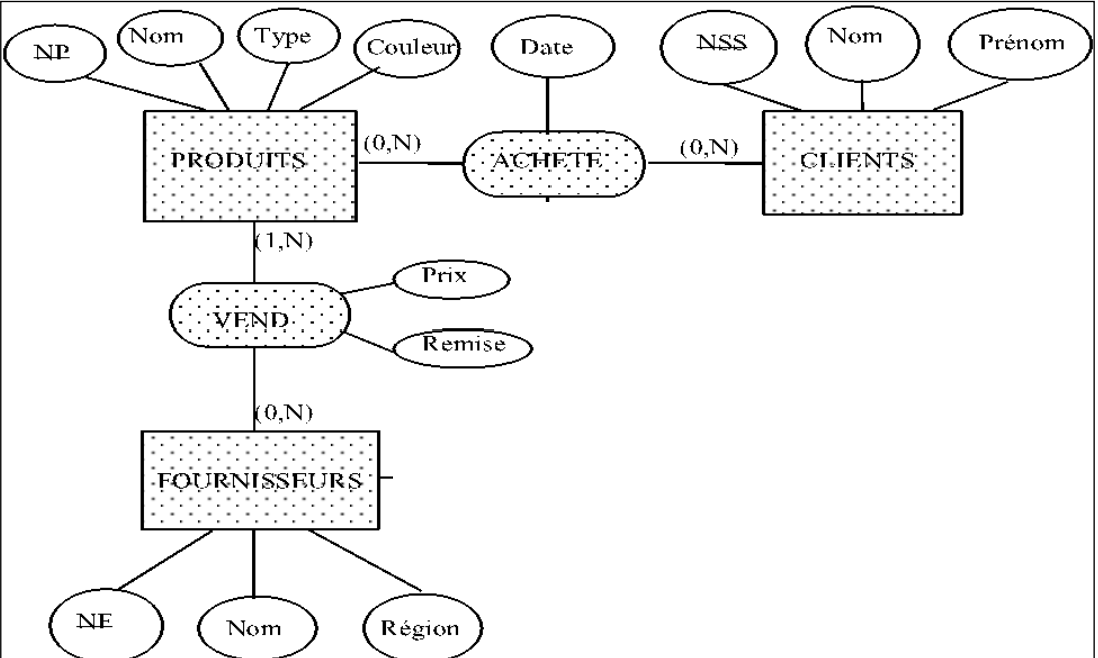
-analyse prédictive

Une base de données (correcte) est un ensemble structuré, cohérent (non contradictoire) et pertinent (représentatif du problème traité et sans redondances) d'informations fortement liées.

Il en découle la nécessité d'appliquer une méthode de conception stricte, rendant possible la conception globale d'une base de données.



II. Modèle entité-association



Entité : Un objet du monde réel qui peut être identifié et que l'on souhaite représenter.

Attribut : Description des propriétés des entités

– *Attribut simple* : attribut ayant une valeur d'un type de base

– *Attribut composé* : attribut constitué d'un groupe d'attributs

– *Attribut multivalué :* attribut pouvant avoir plus d'une valeur

Identifiant ou Clé : identifier d’une façon unique l’entité

– Un identifiant peut être constitué de plusieurs attributs (clé composée)

Association : Les entités sont reliées ensemble par des associations

– Une association peut avoir des attributs (propriétés)

– Elle peut relier plusieurs entités ensemble

– Il est possible de distinguer le rôle d'une entité (elle peut en avoir plusieurs)

Cardinalité (Multiplicity) : elle se note (mini, maxi) sur chaque « branche = » de l’association

– Le nombre d'instance d'une entité pour chaque instance de l'autre

Associations

- Une association doit toujours mettre en relation le même nombre d'entités (il est interdit d'avoir une relation entre A, B et C qui, de temps en temps, ne relierait que A et B)

- Si les attributs d'une association peuvent être mis dans une entité supplémentaire en relation avec l'association, le faire.

- Si un attribut de l'association n'a de sens que pour un sous-ensemble des entités concerné, alors cet attribut fait partie du sous-ensemble concerné et non de l'association complète

*Recensement des informations :*

Il s'agit d'élaborer un dictionnaire des informations à prendre en compte. C'est une démarche sans méthode stricte ni modèle réellement définis : =>on récolte de l'information et on classe.

Deux solutions :

-Discuter avec le client et les intervenants

-Lire les documents

On obtient une liste d’informations qu’il faudra épurer :

-On précisera les polysémies ("masse" signifie "poids" ou "gros marteau" ?); -> On éliminera les synonymies;

-On rencontrera des informations élaborées : il faudra alors énoncer la règle d'élaboration ; on conservera, ou non, ces informations selon l'urgence et la fréquence de leur emploi

-On gardera les informations élémentaires (non élaborées) dans le modèle E/A

*La pratique de la conception :*

• Bien lire et analyser le cahier des charges et comprendre le problème à résoudre

• Identifier les entités, pour chaque entité identifier les attributs et parmi tous ces attributs choisir la clé (l’identifiant unique)

• Identifier les associations, pour chaque association identifier ces attributs

• Identifier les cardinalités sur chaque côté de l’association

• Documenter les significations et conventions

III. Modèle relationnel

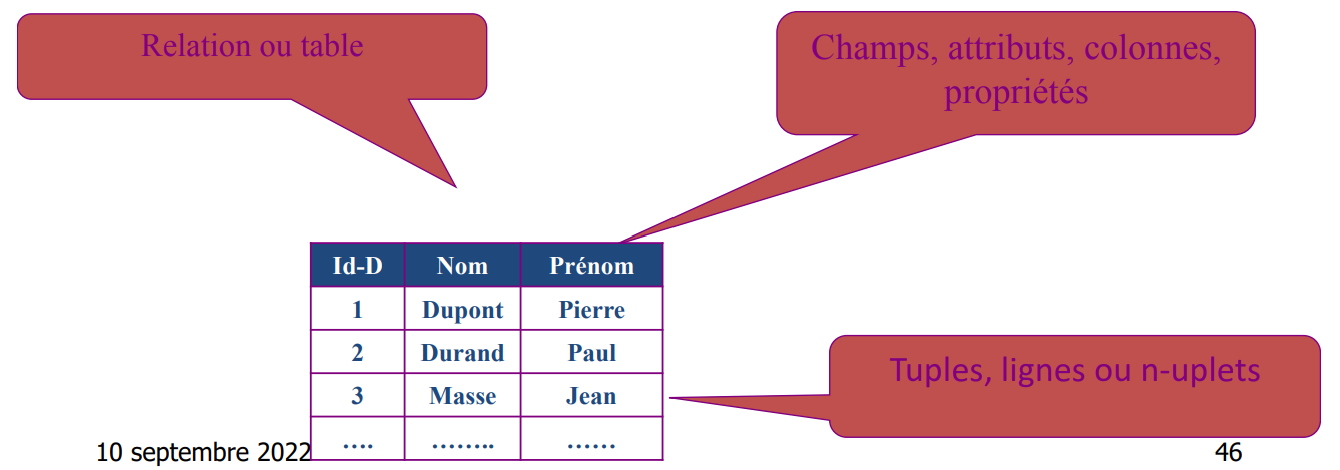
*Le passage au relationnel*

• Implémentation des entités et associations sous forme de tables

• Les attributs correspondent aux colonnes des tables

– nom attribut => nom colonne

– Ensemble de valeurs => domaine



*Remarques :*

-Le nombre d’attributs est appelé : degré

-Le nombre de tuples est appelé : cardinalité

Relation 1,1 -> 1,N :

L’entité en 1,1 récupère la table étrangère #

Relation 1,N -> 1,N :

Création d’une nouvelle entité « verbe » avec les deux clés étrangères #

Relation 1,1 -> 1,1 :

Chaque entité récupère la clé étrangère de l’autre #

*Relation à plus de deux entités*

On procède comme pour le cas à cardinalité (?,n) - (?,n); c-à-d la relation devient une entité-lien et récupère les identifiants des entités concernées.

*Entité faible :*

Il existe des cas où une entité ne peut exister qu'en étroite association avec une autre, et est identifiée relativement à cette autre entité.

*Héritage*

Les sous-classes RESPONSABLE et ACTEUR feront l’objet elles aussi d’entités types, associées à l’entité-type PERSONNE par une relation d’héritage. Les entités-types RESPONSABLE et ACTEUR possèdent les mêmes attributs que les sous-classes

IV. Normalisation

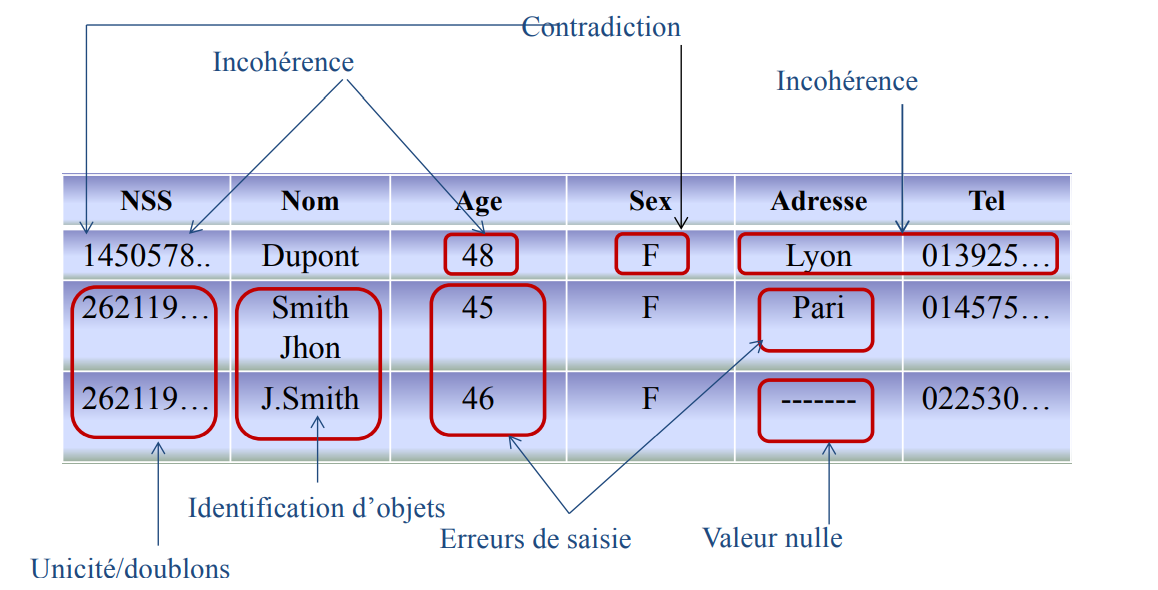
Qualité d’un modèle :

Deux types de conflits sémantiques :

– Conflits liés au schéma : utilisation d’une terminologie différente pour désigner deux concepts identiques

– Conflits liés aux données : La provenance de données de diverses origines, leur saisie à des moments distincts par des personnes différentes qui n’ont pas la même perception du réel, et qui utilisent des conventions différentes

Problème principal = redondance d’information



*Dépendances Fonctionnelles*

– Les DF entre les attributs d’une entité doivent vérifier la règle suivante :

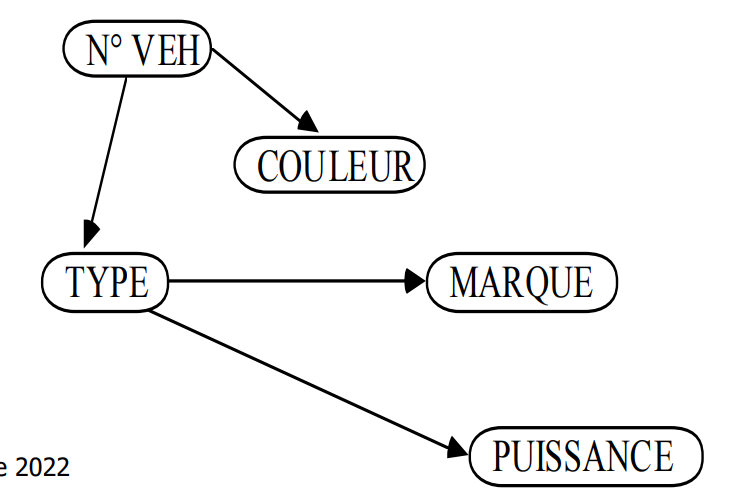
• tous les attributs d’une entité dépendent fonctionnellement et uniquement de l’identifiant

– soit R (A1, A2 … An) un schéma de relation, X et Y des sous-ensembles de A1, A2 …An

– on dit que X ®Y (X détermine Y ou Y dépend fonctionnellement de X) ssi il existe une fonction qui à partir de toute valeur de X détermine une valeur unique de Y

• Formellement :

– ssi quel que soit l’instance r de R, pour tout tuple t1 et t2 de r on a PX(t1) = PX(t2) ==> PY(t1) = PY(t2)



Exemple de graphe DF

*Formes normales*

• Objectifs :

– Définir des règles pour décomposer les relations tout en préservant les DF et sans perdre d'informations, afin de représenter des objets et associations du monde réel

– Éviter les anomalies de mises à jour

– Éviter les réponses erronées

+ 1ère Forme (1NF)

Une relation est en 1NF si tout attribut contient une valeur atomique (unique).

+ 2ème Forme (2NF)

Une relation est en 2NF ssi :

• elle est en 1ère forme

• Aucune partie de la clé ne détermine un autre attribut non-clé

+ 3ème Forme (3NF)

Une relation est en 3NF ssi :

• elle est en 2NF

• Pas de dépendances entre les attributs non-clés

V. Algèbre Relationnelle

Un ensemble d'opérations formelles

• Ces opérations permettent d'exprimer toutes les requêtes sous forme d'expressions algébriques

• Elles sont la base du langage SQL :

– Paraphrase en anglais des expressions relationnelles

– Origine SEQUEL

• Ces opérations se généralisent à l'objet

– Algèbre d'objets complexes

Interrogation de la base de données à travers des opérateurs de type algébriques, c’est-à-dire qui peuvent se composer pour obtenir un résultat :

– Projection

– Restriction

– Union

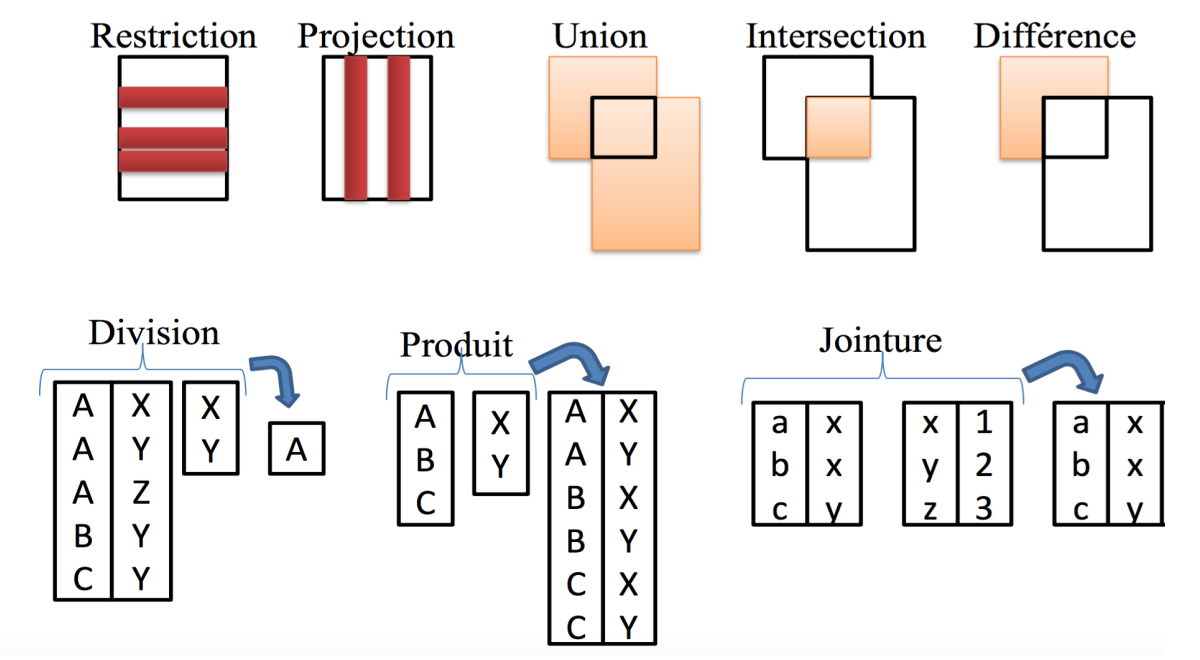
– Différence

– Intersection

– Produit cartésien

– Jointure

– Division



*La projection*

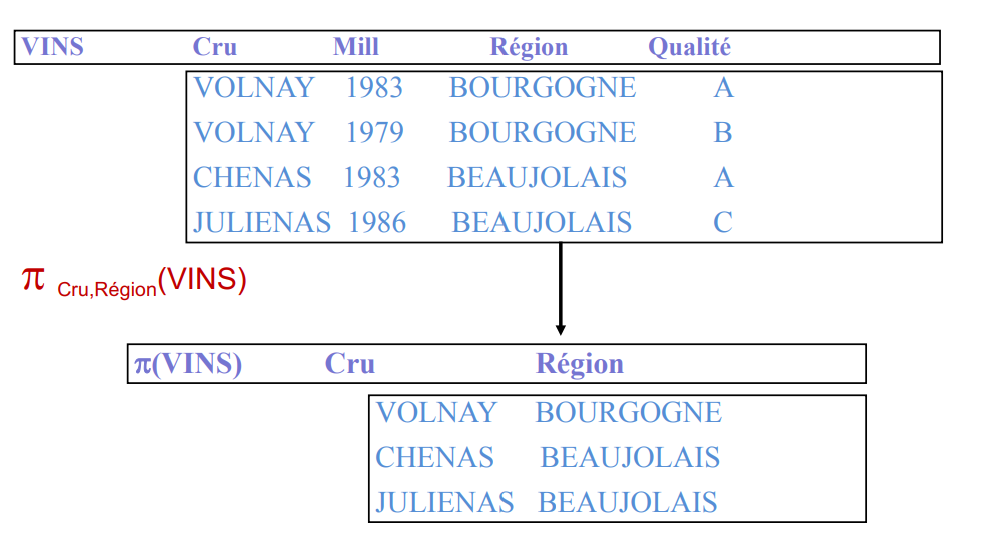
La projection d'une relation R sur un ensemble d'attributs donne une relation ayant pour schéma cet ensemble d'attributs

• Ses tuples sont ceux de la relation R, réduits aux attributs de la projection.

• Elimination des attributs non désirés et suppression des tuples en double

• Relation →Relation notée : π A1, A2, ..., Ap (R) 🡺SELECT

Exemple :



*La restriction*

• Obtention des tuples de R satisfaisant un critère Q

• Relation ->Relation, notée σQ(R) 🡺WHERE

• Q est le critère de qualification de la forme :

– Ai q Valeur

– q = s {=, =, >, <=, !=}

• Il est possible de réaliser des "ou" (union) et des "et" (intersection) de critères simples :

ET (∧) ; OU (∨)

Exemple :

Une image contenant table

Description générée automatiquement

*Restriction/ Projection*

Les opérateurs de restriction et de projection peuvent se combiner pour sélectionner un résultat :

Une image contenant table

Description générée automatiquement

*Opérations ensemblistes*

• Opérations pour des relations de même schéma

– UNION

– INTERSECTION

– DIFFERENCE notée —

• Opérations binaires

– Relation X Relation → Relation

• Extension

– Union externe pour des relations de schémas différents

– Ramener au même schéma avec des valeurs nulles

*Produit cartésien*

• Le produit cartésien de deux relations R1 et R2 est une relation dont le schéma est la concaténation des schémas de R1 et R2

• Ses tuples sont ceux de R1, concaténés chacun à tous les tuples de R2

*Division*

• La division de deux relations R1 et R2, telles que le schéma de R2 est strictement inclus dans celui de R1, est une relation R3 regroupant toutes les parties d’occurrences de la relation R1 qui sont associées à toutes les occurrences de la relaQon R2, noté : R3 = (R1 ÷ R2)

Une image contenant table

Description générée automatiquement

*Jointure*

• Composition des deux relations sur un domaine commun

• Relation X Relation →Relation

– notée

• Critère de jointure

– Attributs de même nom égaux :

• Attribut = Attribut appelée Jointure naturelle

– Comparaison d'attributs :

• Attribut1 Thêta Attribut2 appelée Thêta-jointure

Une jointure est un produit cartésien suivi d'une restriction sur un critère.

Une image contenant texte

Description générée automatiquement

VI. SQL

• Une interface plus simple pour les utilisateurs

– La majorité des manipulations utilisateurs se font via des formulaires !

– Mais rend possible les requêtes libres (ex: analyse)

• Simplifier le développement des applications

– Code plus compact et facile à valider/maintenir

– Indépendance physique et logique des programmes par rapport aux données

– Optimisation automatique et dynamique des requêtes par le SGBD

– Une puissance d’expression exploitable de nb façons

– Sélection, mises à jour, intégrité, droits d’accès, audit …

*Création de table : Syntaxe*

• Syntaxe

CREATE TABLE <relation name>

(<attribute definition>+)

[{PRIMARY KEY | UNIQUE} (<attribute name>+)]

• Exemple

CREATE TABLE RDV (

NumRdv Integer PRIMARY KEY,

DateRDV Date,

NumDoc Integer,

NumPat Integer,

Motif Varchar (200),

FOREIGN KEY (NumDoc) REFERENCES DOC (NumDoc),

FOREIGN KEY (NumPat) REFERENCES PAT (NumPat)

)

*Insertion de données*

• Syntaxe

INSERT INTO <relation name>

[(attribute [,attribute] …)]

{VALUES <value spec.> [, <value spec.>] …| <query spec.>}

• Exemples :

– INSERT INTO DOC VALUES (1, ‘Dupont’, ‘Paris’);

– INSERT INTO DOC (NumDoc, NomDoc) VALUES (2, ‘Toto’);

*Suppression de données*

Syntaxe :

DELETE FROM <relation\_name>

[WHERE <search\_condition>]

• Exemples :

Supprimer tous les docteurs

DELETE FROM DOC

Supprimer le docteur numéro 20

DELETE FROM DOC

WHERE NumDoc = 20

*Modification de données*

• SYNTAXE:

UPDATE <relation\_name>

SET <attribute> = value\_expression

[, <attribute> = value\_expression] …

[WHERE <search condition>];

• EXEMPLES :

Modifier la ville du docteur dont le numéro est 20

UPDATE DOC

SET VilleDoc = ‘Valenton’

WHERE NumDoc = 20

Augmenter de 10% le prix de tous les médicaments dont le nom contient ‘Antibio’

UPDATE MED

SET PRIX = PRIX\*1.1

WHERE NomMed = ‘%Antibio%’

*Interrogation des données*

• Une requête SQL est une paraphrase d'une expression de l'algèbre relationnelle en anglais

• Requête élémentaire

SELECT <liste des attributs projetés Ai>

FROM <liste des relations impliquées Rj>

WHERE <critères de recherche Ci>

• Sémantique du bloc

PROJECTION Ai (

RESTRICTION Ci (

PRODUIT (Rj))

*Structure générale d’une requête SQL*

SELECT [DISTINCT/ALL] expression sur les attributs |\*

FROM table [variable], table [variable], ...

[WHERE condition de sélection /sous-question]

[GROUP BY liste d’attributs]

[HAVING condition de sélection /sous-question]

[ORDER BY liste d’attributs] ;

• Restriction :

–arithmétique (=, <, >, …)

–textuelle (LIKE)

–sur intervalle (BETWEEN)

–sur liste (IN)

• Possibilité de blocs imbriqués par :

IN, EXISTS, NOT EXISTS, ALL, SOME, ANY

Evaluation d’une requête SQL

§ La clause FROM : en faisant le produit cartésien de toutes les relations qui apparaissent dans la requête

§ La clause WHERE : réalise la restriction et les jointures

§ La clause GROUP BY : qui constitue les partitions

§ La clause HAVING : qui sélectionne les partitions désirées

§ La clause SELECT : qui constitue le projection finale

§ La clause ORDER BY : qui trie les tuples du résultat final